

# ECサイト初級

ログイン機能の作成

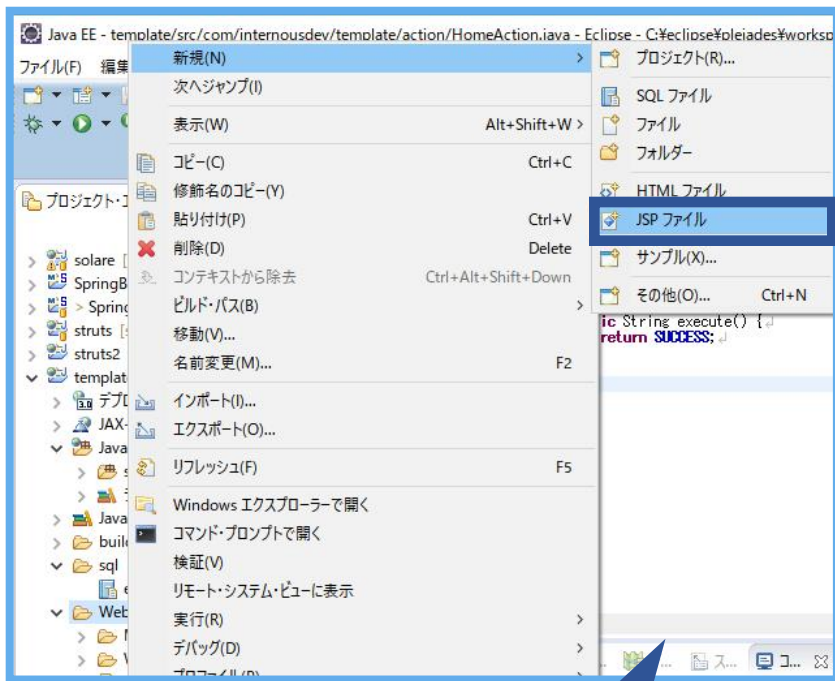
## 3 時間目

## 作成機能一覧

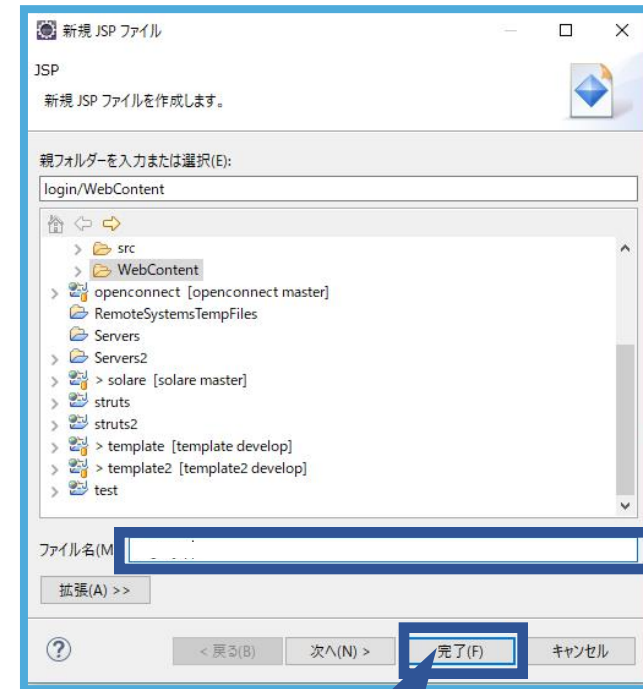
1. ログイン認証機能
2. ユーザー登録機能
3. 商品購入機能
4. 購入履歴機能

# JSPファイルの作成

## 1 home.jsp



① 「プロジェクト」「WebContent」を選択し、「右クリック」「新規」「JSPファイル」を選択します。



② 「名前(M):」欄に「home.jsp」を入力し、完了ボタンをクリックします。

# home.jspの作成

③ 以下の内容を写経します。

home.jsp(jspファイル)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="s" uri="/struts-tags"%>
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <meta http-equiv="imagetoolbar" content="no" />
    <meta name="description" content="" />
    <meta name="keywords" content="" />
    <title>Home画面</title>

    <style type="text/css">
/* =====TAG LAYOUT===== */
```

次へ続きます。

前の続きです。

```
body {  
    margin:0;  
    padding:0;  
    line-height:1.6;  
    letter-spacing:1px;  
    font-family:Verdana, Helvetica, sans-serif;  
    font-size:12px;  
    color:#333;  
    background:#fff;  
}  
  
table {  
    text-align:center;  
    margin:0 auto;  
}  
/* =====ID LAYOUT===== */  
#top {  
    width:780px;  
    margin:30px auto;
```

次へ続きます。

前の続きです。

```
border:1px solid #333;
}

#header {
width: 100%;
height: 80px;
background-color: black;
}

#main {
width: 100%;
height: 500px;
text-align: center;
}

#footer {
width: 100%;
height: 80px;
background-color: black;
clear:both;
}
```

次へ続きます。

前の続きです。

```
</style>
</head>
<body>
  <div id="header">
    <div id="pr">
    </div>
  </div>
  <div id="main">
    <div id="top">
      <p>Home</p>
    </div>
    <div>
      <s:form action="HomeAction">
        <s:submit value="商品購入"/>
      </s:form>
    </div>
  </div>
  <div id="footer">
    <div id="pr">
```

次へ続きます。

前の続きです。

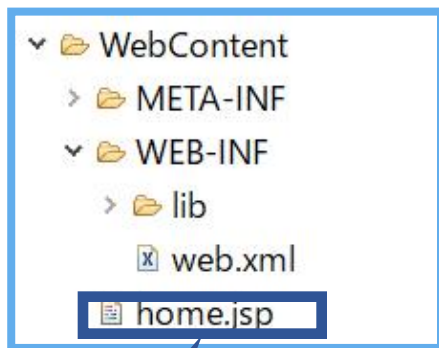
```
</div>  
</div>  
</body>  
</html>
```



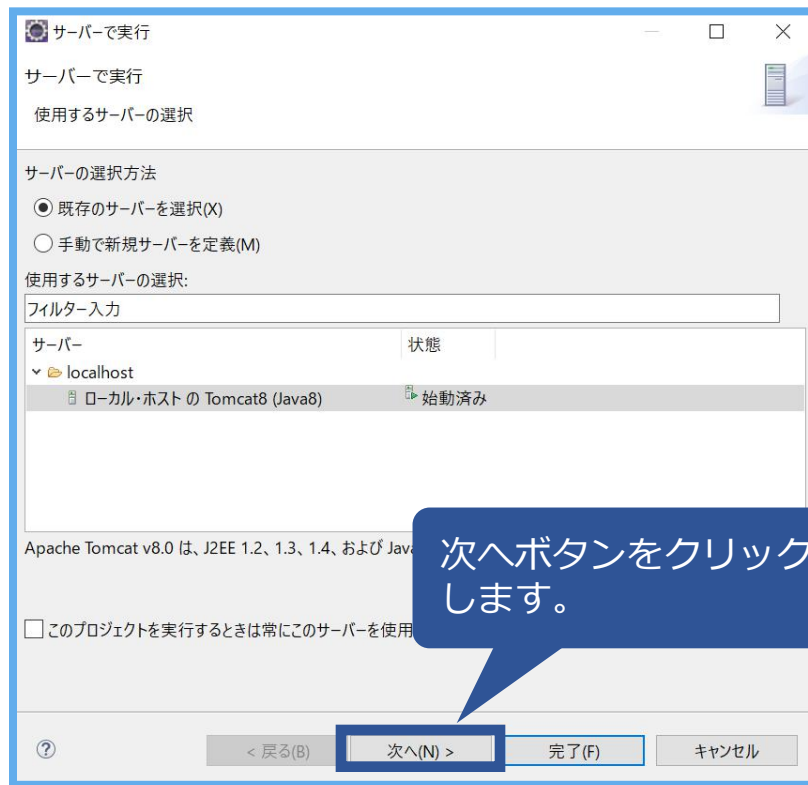
# jspファイルの実行方法

## 解説

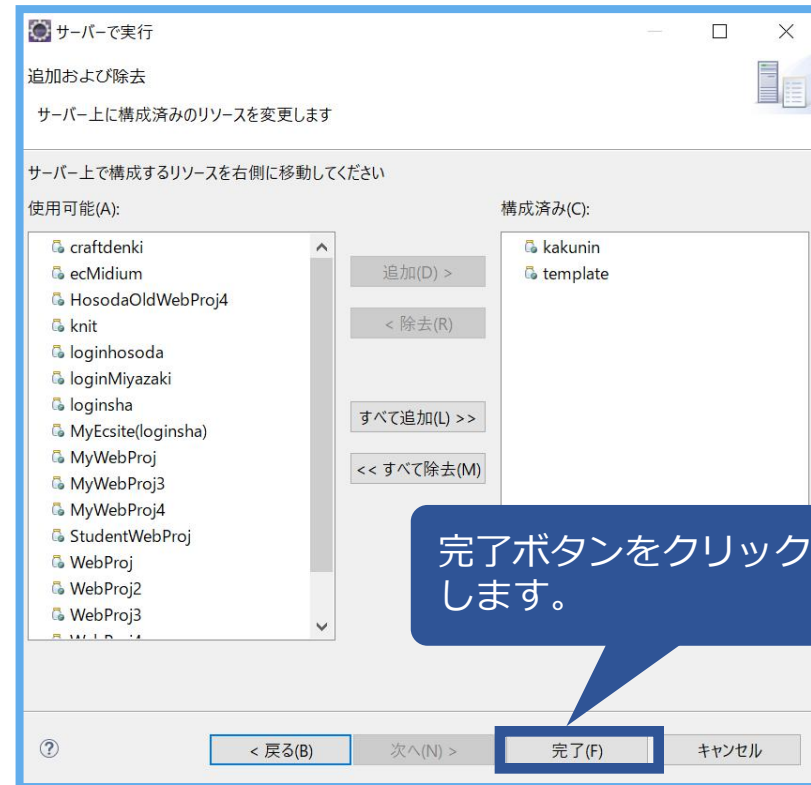
必要なActionファイルを記述していない段階でjspファイルの実行結果を確かめるには、jspファイルだけを単独で実行します。  
※ templateプロジェクト自体を実行すると、必要なActionファイルが無い場合、途中でエラーになってしまいます。



該当のjspファイルを  
右クリックします。



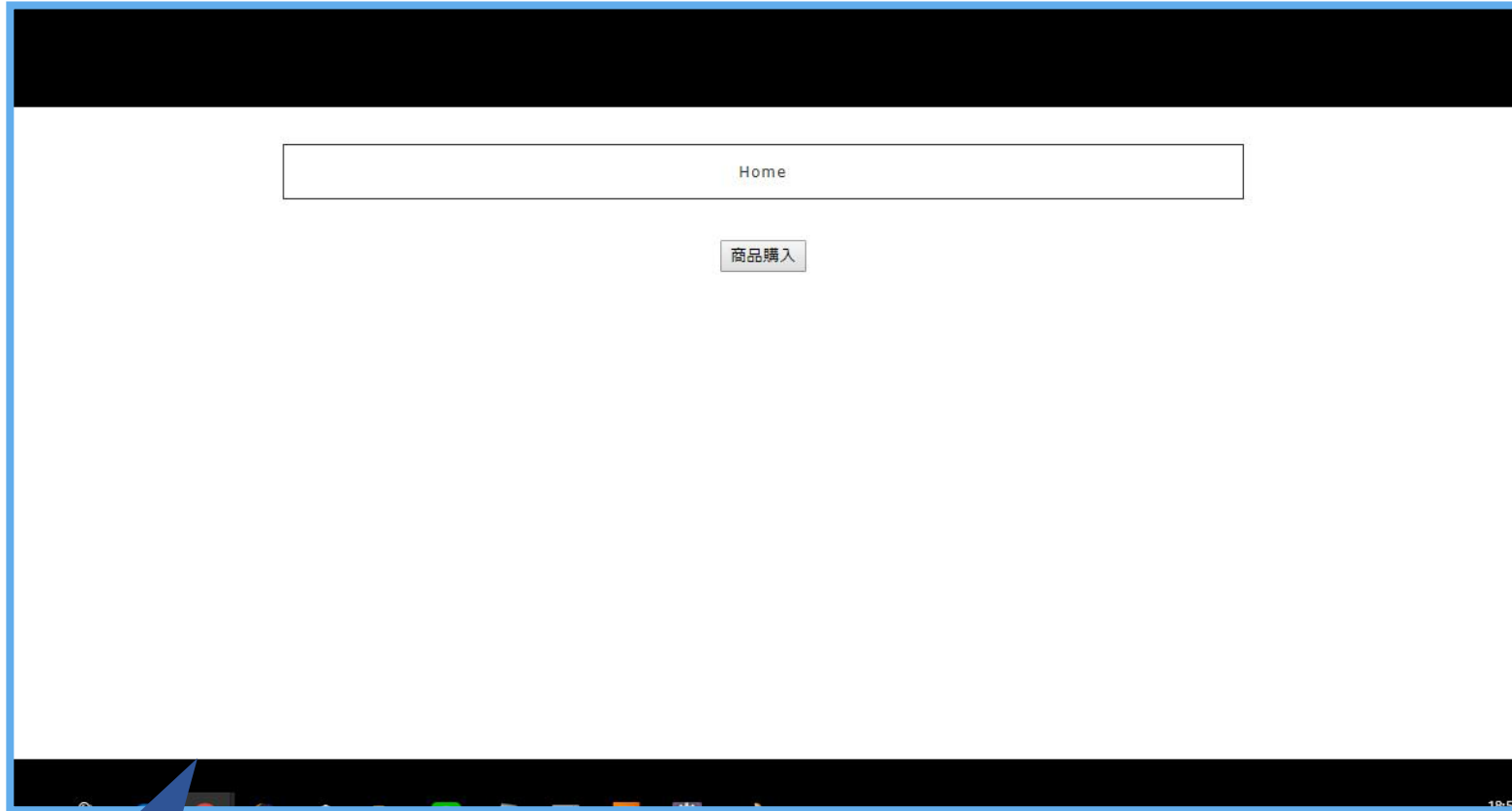
次へボタンをクリック  
します。



完了ボタンをクリック  
します。

# home.jspの作成

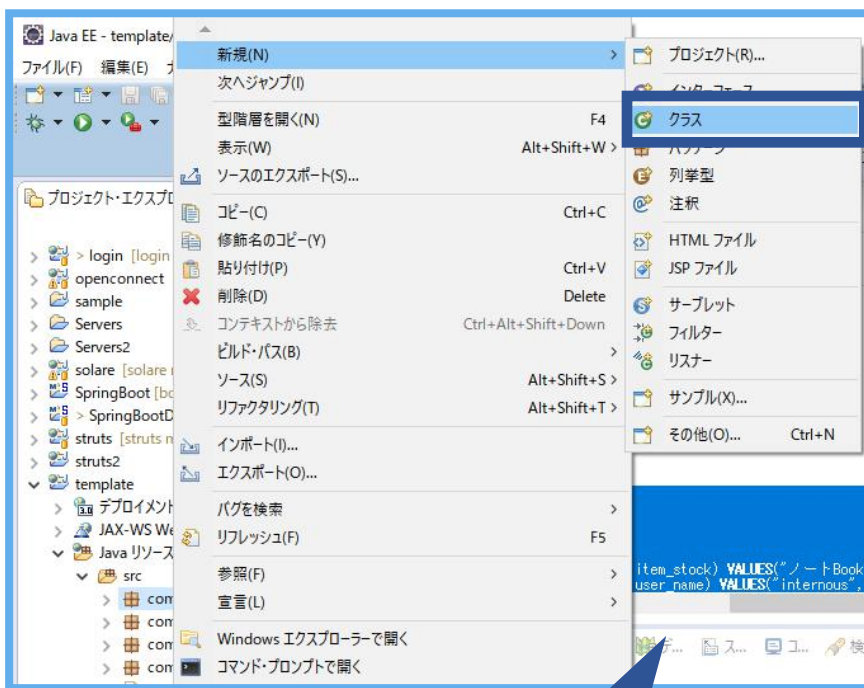
## home.jspの画面



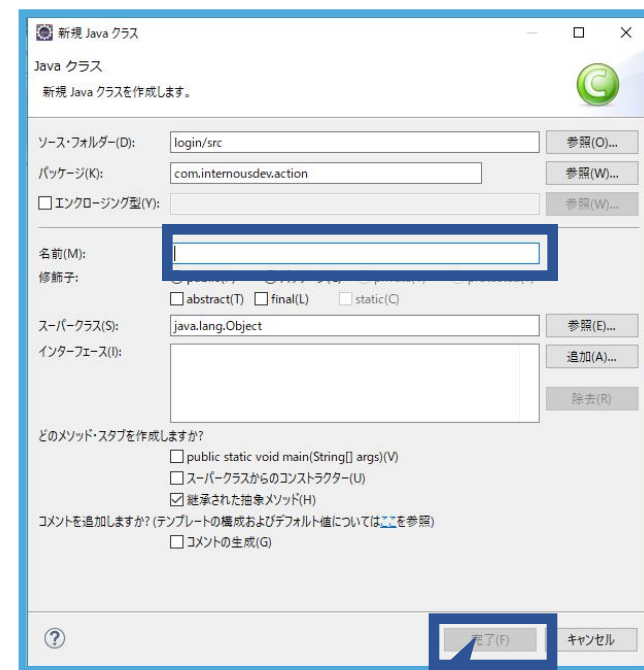
上記の画面が表示できれば成功です。

# HomeAction.javaの作成

## 2 HomeAction.java



① 「com.internousdev.template.action」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「HomeAction」を入力し、完了ボタンをクリックします。

# HomeAction.javaの作成

③ 以下の内容を写経します。

HomeAction(javaファイル)

```
package com.internousdev.template.action;

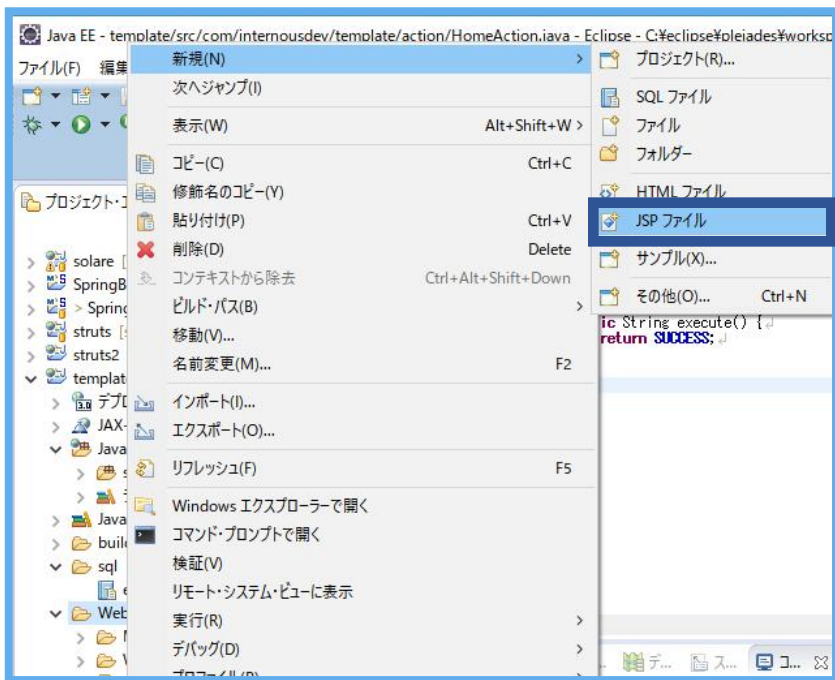
import com.opensymphony.xwork2.ActionSupport;

public class HomeAction extends ActionSupport {
    public String execute() {
        return SUCCESS;
    }
}
```

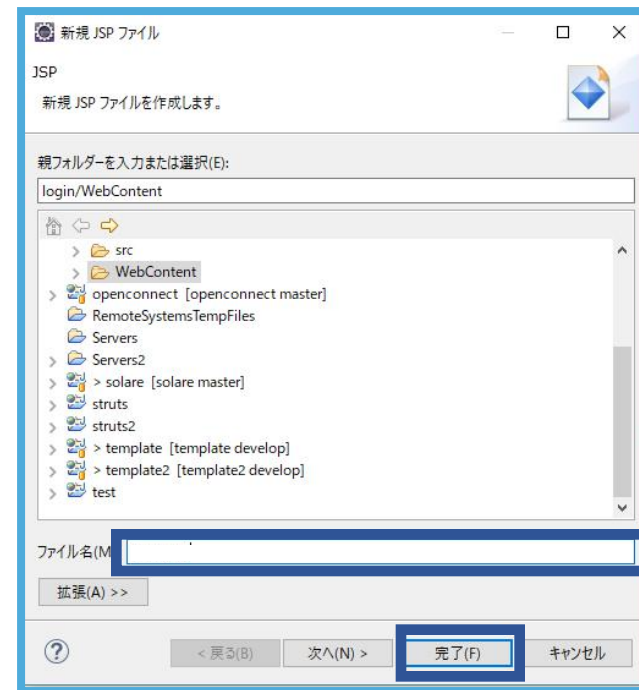
「execute」メソッドを実行した後は  
"success"文字列を返します。

# login.jspの作成

## 3 login.jsp



① 「プロジェクト」「WebContent」を選択し、「右クリック」「新規」「JSPファイル」を選択します。



② 「名前(M):」欄に「login.jsp」を入力し、完了ボタンをクリックします。

# login.jspの作成

③ 以下の内容を写経します。

login.jsp(jspファイル)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="s" uri="/struts-tags"%>
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <meta http-equiv="imagetoolbar" content="no" />
    <meta name="description" content="" />
    <meta name="keywords" content="" />
    <title>Login画面</title>

    <style type="text/css">
/* =====TAG LAYOUT===== */
```

次へ続きます。

前の続きです。

```
body {  
    margin:0;  
    padding:0;  
    line-height:1.6;  
    letter-spacing:1px;  
    font-family:Verdana, Helvetica, sans-serif;  
    font-size:12px;  
    color:#333;  
    background:#fff;  
}  
  
table {  
    text-align:center;  
    margin:0 auto;  
}  
/* =====ID LAYOUT===== */  
#top {  
    width:780px;  
    margin:30px auto;
```

次へ続きます。

前の続きです。

```
border:1px solid #333;
}

#header {
width: 100%;
height: 80px;
background-color: black;
}

#main {
width: 100%;
height: 500px;
text-align: center;
}

#footer {
width: 100%;
height: 80px;
background-color: black;
clear:both;
}
```

次へ続きます。



前の続きです。

```
</style>
</head>
<body>
  <div id="header">
    <div id="pr">
    </div>
  </div>
  <div id="main">
    <div id="top">
      <p>Login</p>
    </div>
    <div>
      <h3>商品を購入する際にはログインをお願いします。</h3>
      <s:form action="LoginAction">
        <s:textfield name="loginUserId"/>
        <s:password name="loginPassword"/>
        <s:submit value="ログイン"/>
      </s:form>
      <br/>
    </div>
  </div>
</body>
</html>
```

次へ続きます。

# login.jspの作成

前の続きです。

```
        <div>
            <span>新規ユーザー登録は
                <a href='<s:url action="UserCreateAction" />'>こちら</a>
            </span>
        </div>
    </div>
    </div>
    <div id="footer">
        <div id="pr">
        </div>
    </div>
</body>
</html>
```

# login.jspの作成

## 解説

必要なActionファイルを記述していない段階でjspファイルの実行結果を確認するには、jspファイルだけを単独で実行します。  
※ 「template」プロジェクト自体を実行すると、必要なActionファイルが無い場合、途中でエラーになってしまいます。

## login.jspの画面

Login

商品を購入する際にはログインをお願いします。

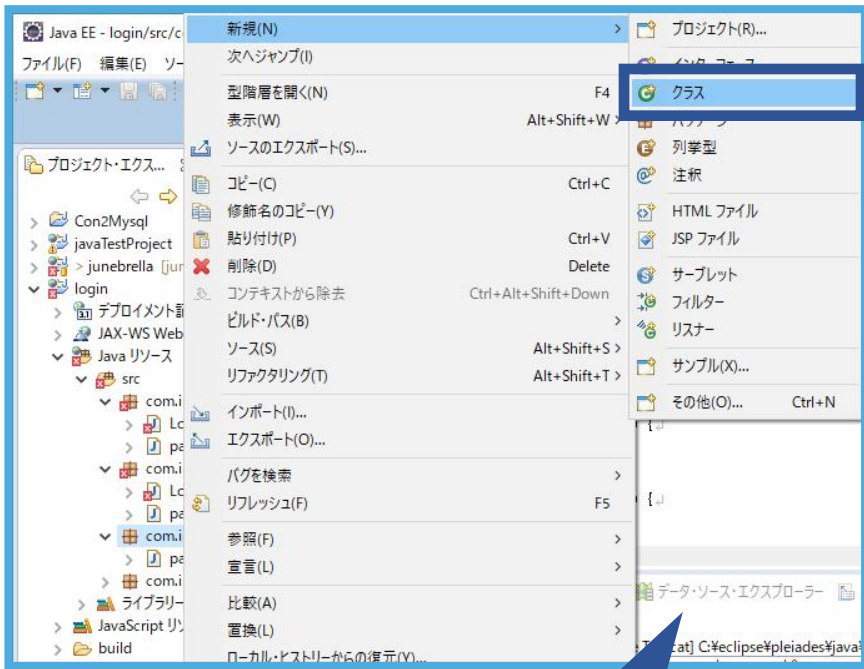
ログイン

新規ユーザー登録は[こちら](#)

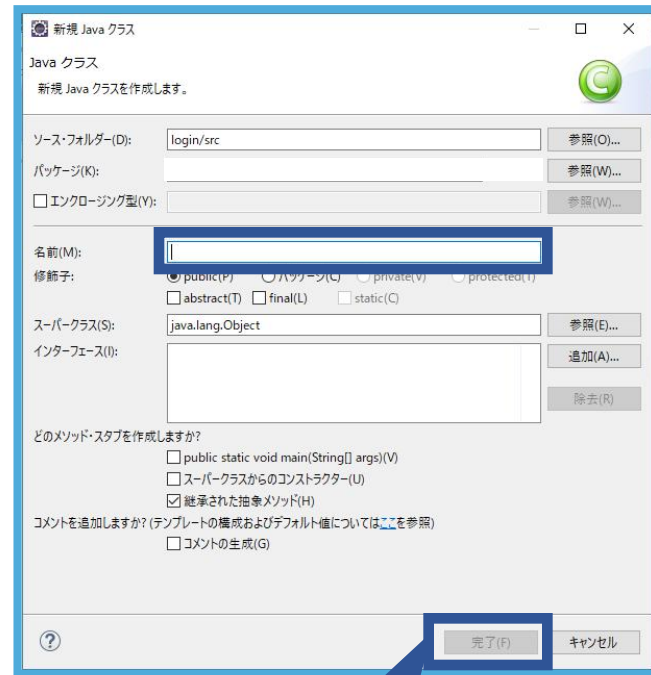
上記の画面が表示できれば成功です。

# LoginDTO.javaの作成

## 4 LoginDTO.java



① 「com.internousdev.template.dto」を選択し、クリック「新規」「クラス」を選択します。



② 「名前(M):」欄に「LoginDTO」を入力し、完了ボタンをクリックします。

# LoginDTO.javaの作成

③ 以下の内容を写経します。

LoginDTO(javaファイル)

```
package com.internousdev.template.dao;

public class LoginDTO {

    private String loginId;
    private String loginPassword;
    private String userName;
    private boolean loginFlg = false;

    public String getLoginId() {
        return loginId;
    }

    public void setLoginId(String loginId) {
        this.loginId = loginId;
    }
}
```

次へ続きます。

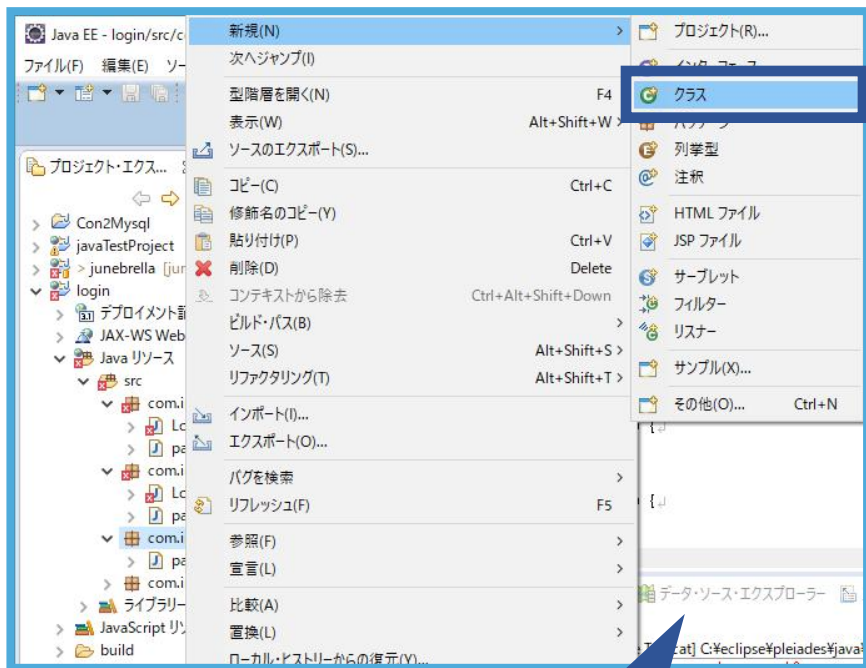
前の続きです。

```
public String getLoginPassword() {  
    return loginPassword;  
}  
public void setLoginPassword(String loginPassword) {  
    this.loginPassword = loginPassword;  
}  
public String getUsername() {  
    return userName;  
}  
public void setUsername(String userName) {  
    this.userName = userName;  
}  
public boolean getLoginFlg() {  
    return loginFlg;  
}  
public void setLoginFlg(boolean loginFlg) {  
    this.loginFlg = loginFlg;  
}  
}
```

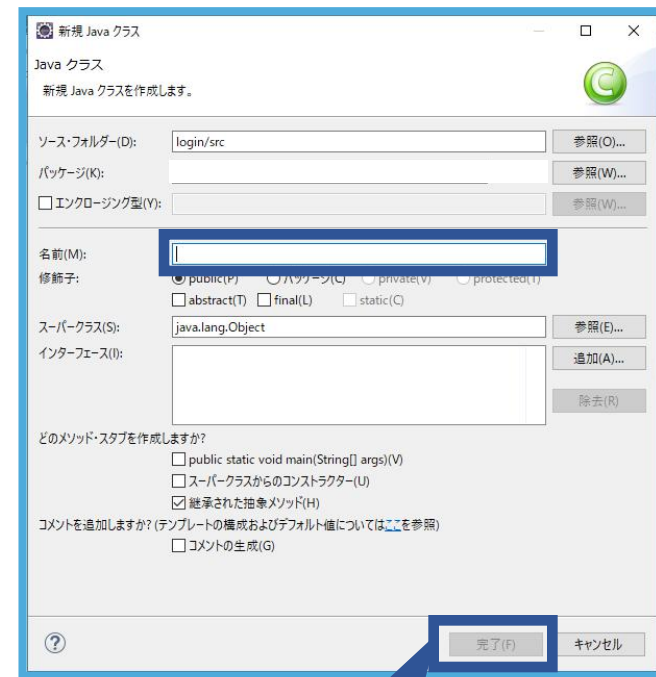
次へ続きます。

# LoginDAO.javaの作成

## 5 LoginDAO.java



① 「com.internousdev.template.dao」を選択し、右クリック「新規」「クラス」を選択します。



② 「名前(M):」欄に「LoginDAO」を入力し、完了ボタンをクリックします。

# LoginDAO.javaの作成

③ 以下の内容を写経します。

LoginDAO(javaファイル)

```
package com.internousdev.template.dao;
```

```
import java.sql.Connection;
```

```
import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet;
```

```
import com.internousdev.template.dto.LoginDTO;
```

```
import com.internousdev.template.util.DBConnector;
```

```
public class LoginDAO {
```

```
    public LoginDTO getLoginUserInfo(String loginUserId, String loginPassword) {
```

```
        DBConnector dbConnector = new DBConnector();
```

```
        Connection connection = dbConnector.getConnection();
```

```
        LoginDTO loginDTO = new LoginDTO();
```

```
        String sql = "SELECT * FROM login_user_transaction where login_id = ? AND
```

```
login_pass = ?";
```

メソッド名は処理内容を分かりやすくするために「getLoginUserInfo」にします。

次へ続きます。



前の続きです。

```
try {  
    PreparedStatement preparedStatement = connection.prepareStatement(sql);  
  
    preparedStatement.setString(1, loginUserId);  
    preparedStatement.setString(2, loginPassword);  
  
    ResultSet resultSet = preparedStatement.executeQuery();  
  
    if(resultSet.next()) {  
        loginDTO.setLoginId(resultSet.getString("login_id"));  
        loginDTO.setLoginPassword(resultSet.getString("login_pass"));  
        loginDTO.setUserName(resultSet.getString("user_name"));  
  
        if(!(resultSet.getString("login_id").equals(null))) {  
            loginDTO.setLoginFlg(true);  
        }  
    }  
}
```

次へ続きます。

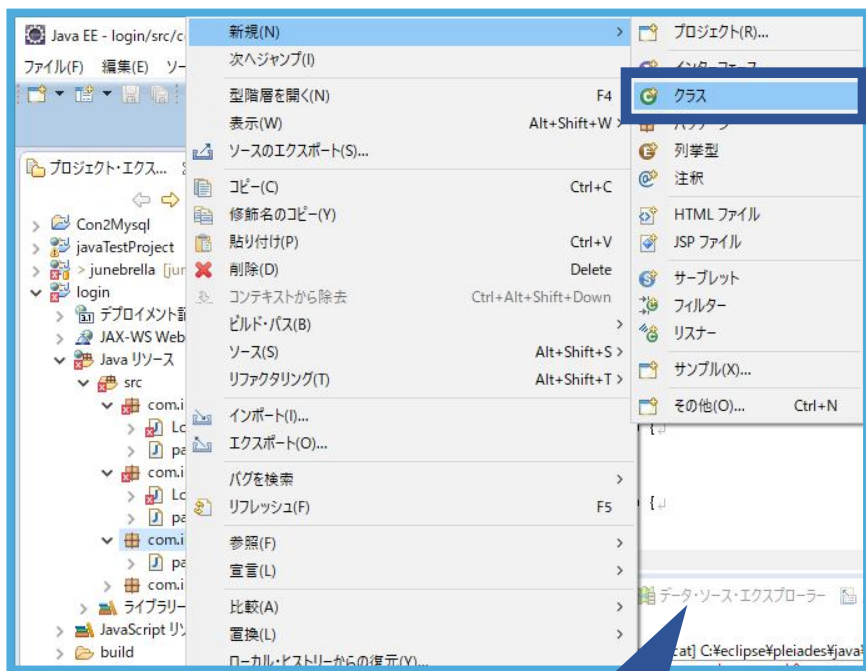
# LoginDAO.javaの作成

前の続きです。

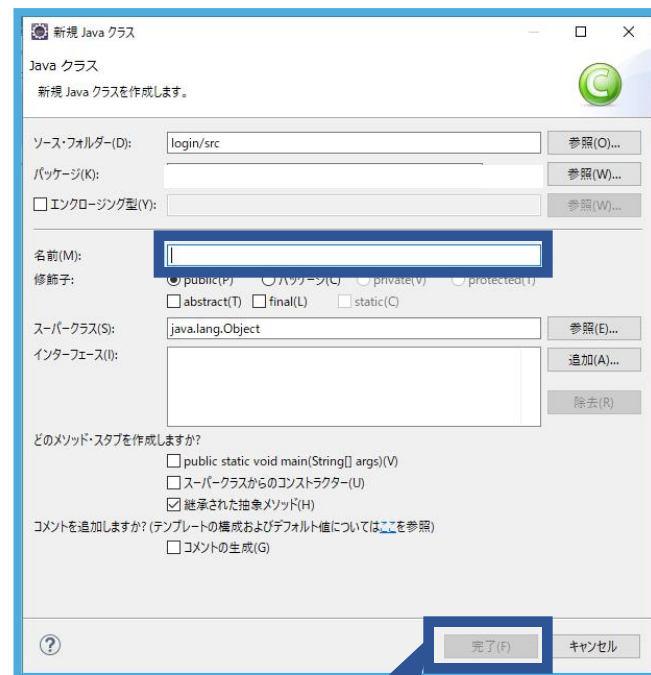
```
    }  
    } catch(Exception e) {  
        e.printStackTrace();  
    }  
    return loginDTO;  
}  
  
}
```

# BuyItemDTOの作成

## 6 DTOクラス



① 「com.internousdev.template.dto」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「BuyItemDTO」を入力し、完了ボタンをクリックします。

# BuyItemDTOの作成

BuyItemDTO(javaファイル)

③ 以下の内容を写経します。

```
package com.internousdev.template.dto;
```

```
public class BuyItemDTO {  
    private int id;  
    private String itemName;  
    private String itemPrice;
```

テーブルカラムに対応したフィールド変数を宣言します。

```
    public String getItemName() {  
        return itemName;  
    }
```

フィールド変数に対応したGetter Setterを定義します。

```
    public void setItemName(String itemName) {  
        this.itemName = itemName;  
    }
```

```
    public String getItemPrice() {  
        return itemPrice;  
    }
```

次へ続きます。

# BuyItemDTOの作成

前の続きです。

```
public void setItemPrice(String itemPrice) {  
    this.itemPrice = itemPrice;  
}
```

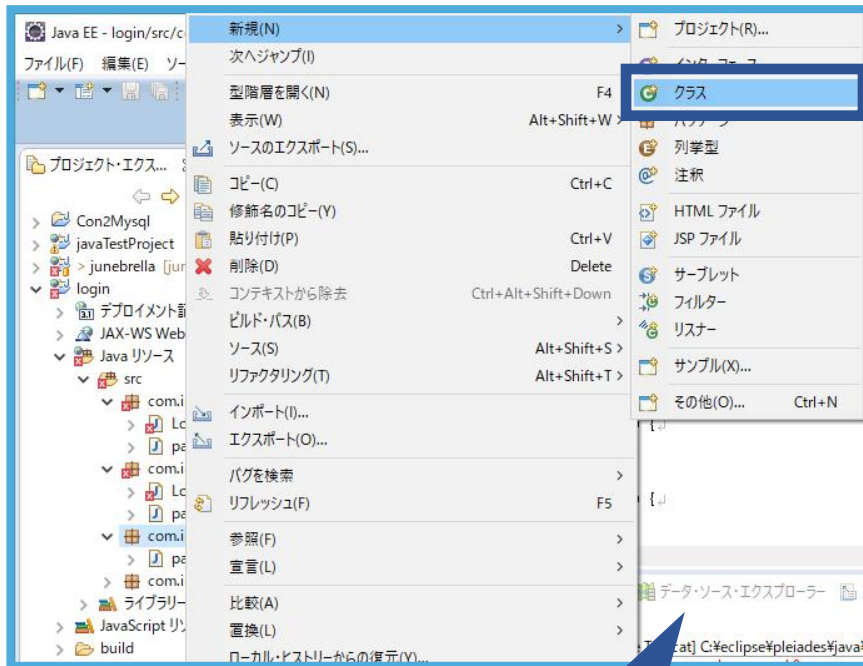
```
public int getId() {  
    return id;  
}
```

```
public void setId(int id) {  
    this.id = id;  
}
```

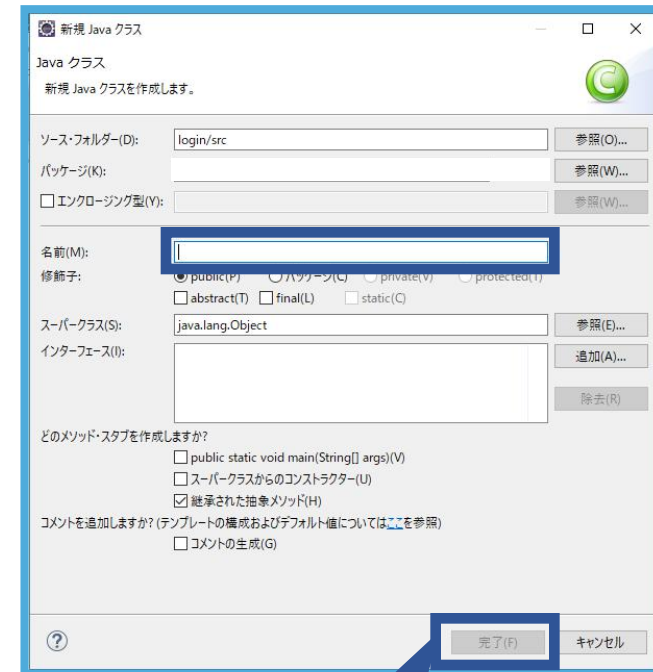
```
}
```

# BuyItemDAO.javaの作成

## 7 BuyItemDAO.java



① 「com.internousdev.template.dao」を選択し、右クリック「新規」「クラス」を選択します。



② 「名前(M):」欄に「BuyItemDAO」を入力し、完了ボタンをクリックします。

# BuyItemDAO.javaの作成

③ 以下の内容を写経します。

BuyItemDAO(javaファイル)

```
package com.internousdev.template.dao;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import com.internousdev.template.dto.BuyItemDTO;
import com.internousdev.template.util.DBConnector;
```

```
public class BuyItemDAO {
```

```
    public BuyItemDTO getBuyItemInfo() {
```

```
        DBConnector dbConnector = new DBConnector();
```

```
        Connection connection = dbConnector.getConnection();
```

```
        BuyItemDTO buyItemDTO = new BuyItemDTO();
```

```
        String sql = "SELECT id, item_name, item_price FROM item_info_transaction";
```

メソッド名は処理内容を分かりやすく  
するために「getBuyItemInfo」にします。

次へ続きます。

# BuyItemDAO.javaの作成

前の続きです。

```
try {
    PreparedStatement preparedStatement = connection.prepareStatement(sql);
    ResultSet resultSet = preparedStatement.executeQuery();

    if(resultSet.next()) {
        buyItemDTO.setId(resultSet.getInt("id"));
        buyItemDTO.setItemName(resultSet.getString("item_name"));
        buyItemDTO.setItemPrice(resultSet.getString("item_price"));
    }

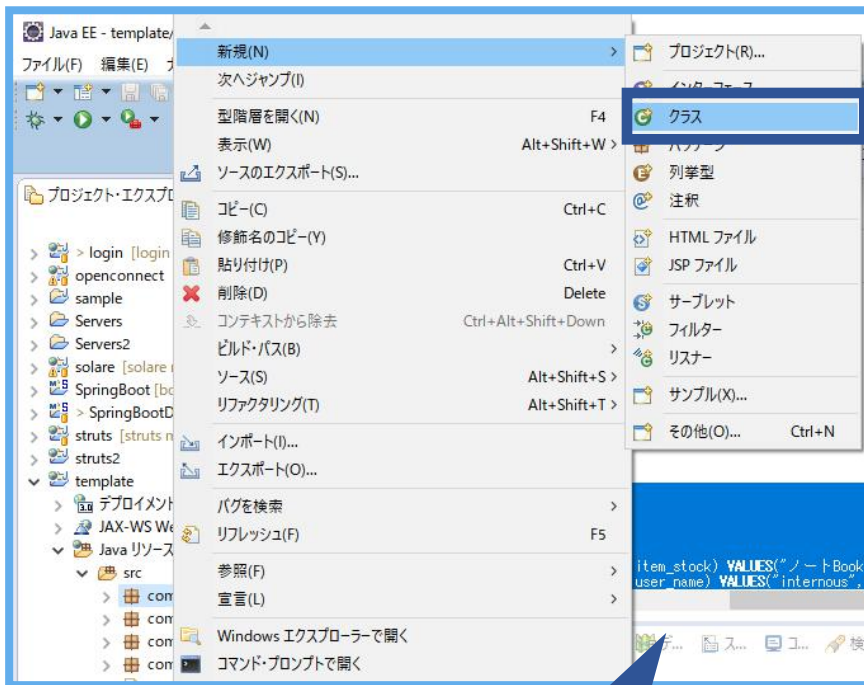
} catch (Exception e) {
    e.printStackTrace();
}

return buyItemDTO;
}
```

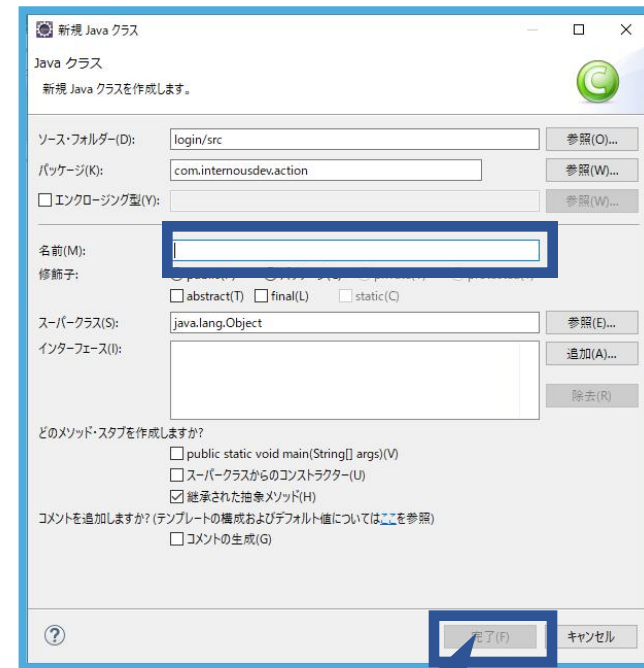


# LoginAction.javaの作成

## 8 LoginAction.java



① 「com.internousdev.template.action」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「LoginAction」を入力し、完了ボタンをクリックします。

# LoginAction.javaの作成

③ 以下の内容を写経します。

LoginAction (javaファイル)

```
package com.internousdev.template.action;
import java.util.Map;
import org.apache.struts2.interceptor.SessionAware;
import com.internousdev.template.dao.BuyItemDAO;
import com.internousdev.template.dao.LoginDAO;
import com.internousdev.template.dto.BuyItemDTO;
import com.internousdev.template.dto.LoginDTO;
import com.opensymphony.xwork2.ActionSupport;

public class LoginAction extends ActionSupport implements SessionAware {

    private String loginUserId;
    private String loginPassword;
    private String result;
    public Map<String, Object> session;
```

# LoginActionの作成

前の続きです。

```
public String execute() {  
    LoginDAO loginDAO = new LoginDAO();  
    LoginDTO loginDTO = new LoginDTO();  
    BuyItemDAO buyItemDAO = new BuyItemDAO();
```

入力値からユーザー情報の検索を行います。  
ログイン認証が成功した場合、次の画面で  
「商品情報」が必要なため商品情報を取得します。

```
    result = ERROR;  
    loginDTO = loginDAO.getLoginUserInfo(loginUserId, loginPassword);  
    session.put("loginUser", loginDTO);
```

```
    if(((LoginDTO) session.get("loginUser")).getLoginFlg()) {  
        result = SUCCESS;  
        BuyItemDTO buyItemDTO = buyItemDAO.getBuyItemInfo();  
  
        session.put("login_user_id", loginDTO.getLoginId());  
        session.put("id", buyItemDTO.getId());  
        session.put("buyItem_name", buyItemDTO.getItemName());  
        session.put("buyItem price", buyItemDTO.getItemPrice());
```

次へ続きます。

# LoginAction.javaの作成

前の続きです。

```
        return result;
    }
    return result;
}

public String getLoginUserId() {
    return loginUserId;
}

public void setLoginUserId(String loginUserId) {
    this.loginUserId = loginUserId;
}

public String getLoginPassword() {
    return loginPassword;
}
```

次へ続きます。

# LoginAction.javaの作成

前の続きです。

```
public void setLoginPassword(String loginPassword) {  
    this.loginPassword = loginPassword;  
}
```

```
@Override  
public void setSession(Map<String, Object> session) {  
    this.session = session;  
}
```

```
}
```

# ログイン認証機能の確認

## buyItem.jsp/home.jspの画面

BuyItem

商品名      NoteBook  
値段        100円  
在庫        1 ▼  
支払い方法 ☒ 現金払い ☐ クレジットカード  
  
前画面に戻る場合は[こちら](#)

正しいID・パスワードを入力した場合

Home

間違ったID・パスワードを入力した場合

templateプロジェクトを実行し、userCreate画面の登録ボタンから上記の上記2パターンの画面が表示できれば成功です。